**IN THE UNITED STATES PATENT & TRADEMARK OFFICE**

| | | |
|---|---|---|
| In re application of Rosario Gennaro | | July 6, 2006 |
| Serial Nbr: | 09/753,727 | |
| Filed: | January 3, 2001 | |
| For: | Technique for Efficiently Generating Pseudo-Random Bits | |
| Art Unit: | 2131 | |
| Examiner: | Matthew T. Henning | |

# APPELLANT'S BRIEF ON APPEAL

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P. O. Box 1450
Alexandria, VA 22313-1450

Sir:

This is an Appeal seeking reversal of the decision of the Primary Examiner, finally rejecting all current claims of the subject patent application.

## 1) REAL PARTY IN INTEREST

The real party in interest is the Assignee, International Business Machines Corporation ("IBM").

## 2) RELATED APPEALS AND INTERFERENCES

Appellant, the Appellant's legal representative, and the assignee, have no personal knowledge of any other appeals or interferences which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

## 3) STATUS OF CLAIMS

Claims 1 - 2, 6 - 7, 9 - 14, 18 - 19, 21 - 26, 30, 32, 34 - 37, 39 - 40, 44, and 47 - 52 stand rejected. Claims 3 - 5, 8, 15 - 17, 20, 27 - 29, 31, 33, 38, 41 - 43, and 45 - 46 have been cancelled from the application without prejudice. Claims 1 - 2, 6 - 7, 9 - 14, 18 - 19, 21 - 26, 30, 32, 34 - 37, 39 - 40, 44, and 47 - 52 are under appeal.

## 4) STATUS OF AMENDMENTS

No Amendments were filed after receiving the Final Rejection mailed on February 15, 2006.

## 5) SUMMARY OF CLAIMED SUBJECT MATTER

1. Appellant's independent Claim 1 specifies elements of "... providing an input value comprising C random bits" (Claim 1, lines 3 - 4); "... generating an output sequence comprising N pseudo-random bits using the provided C-bit input value as a short exponent x of a 1-way function G**x mod P that comprises modular exponentiation modulo a safe N-bit prime number

P, wherein a base G of the modular exponentiation is a fixed generator value" (Claim 1, lines 5 - 8); "... separating the N bits of the generated N-bit output sequence into a C-bit portion and an (N-C)-bit portion" (Claim 1, lines 9 - 10); and "... using the C-bit portion of the generated N-bit output sequence as the provided input value for a next iteration of the computer-readable program code means for generating while using the (N-C)-bit portion of the generated N-bit output sequence as pseudo-random output bits, until a desired number of pseudo-random output bits have been generated" (Claim 1, lines 11 - 15). Independent Claims 13, 25, and 39 specify similar limitations. In other words, an N-bit output is generated, and of these N bits, "C" of them are used as input to a next iteration while the other (N - C) bits are output for forming a pseudo-random value (which may be used, for example, as input to an encryption operation).

2. Appellant's independent Claim 52 specifies elements of "providing an N-bit input value in which (N-C) uppermost contiguous ones of the bits are all set to zeroes and in which C lowermost contiguous ones of the bits are random" (Claim 52, lines 3 - 4); "generating an output sequence comprising N pseudo-random bits using the provided N-bit input value as an effectively-short, C-bit exponent x of a 1-way function G**x mod P that comprises modular exponentiation modulo a safe N-bit prime number P, wherein a base G of the modular exponentiation is a fixed generator value" (Claim 52, line 5 - 8); "separating the N bits of the generated N-bit output sequence into a C-bit portion and an (N-C)-bit portion" (Claim 52, line 9 - 10); "creating a new N-bit input value in which the (N-C) uppermost contiguous ones of the bits are all set to zeroes and in which the lowermost C contiguous ones of the bits are set to the C-bit portion" (Claim 52, lines 11 - 13); and "using the new N-bit input value as the provided input value for a next iteration

of the generating step while using the (N-C)-bit portion of the generated N-bit output sequence as pseudo-random output bits, until a desired number of pseudo-random output bits have been generated" (Claim 52, lines 14 - 17).

3. Dependent Claims 2, 14, 26, and 40 specify "wherein the 1-way function is based upon an assumption known as "the discrete logarithm with short exponent" assumption".

4. Dependent Claims 6, 18, 30, and 44 specify "wherein C = 160 and N = 1024" whereas Dependent Claims 7, 19, and 32 specify "wherein C is greater than or equal to 160 and N is greater than or equal to 1024 " (or "at least", rather than "greater than or equal to", in Claim 19).

5. Dependent Claims 9, 21, 34, and 47 specify "wherein the (N-C)-bit portion is concatenated to pseudo-random output bits previously generated by the [generator]".

6. Dependent Claims 10, 22, and 35 specify "wherein the (N-C)-bit portion is a contiguous group of (N-C) bits from the generated N-bit output sequence". Dependent Claims 11, 23, and 36 specify "wherein the (N-C)-bit portion is a non-contiguous group of (N-C) bits from the generated N-bit output sequence".

7. Dependent Claims 12, 24, and 37 specify "further comprising ... using the desired number of generated pseudo-random bits as input to an encryption operation".

8. Dependent Claims 48 - 51 specify “wherein N is greater than or equal to (C * 6)”.

9. Independent Claims 1, 13, and 39 and dependent Claims 12 and 24 include means plus function terminology. Structure, material, or acts supporting this terminology are described in Appellant’s specification, as will now be described.

10. With regard to the “means for providing” element of independent Claims 1, 13, and 39, the text on p. 6, lines 10 - 12 of Appellant’s specification describes use of hardware for generating random seeds (e.g., the “C” random bits referred to in the “means for providing”), and p. 15, lines 6 - 7 refer to an alternative that comprises implementing the present invention in hardware (or a combination of hardware and software). For the “means for generating” element of independent Claims 1, 13,and 39, refer to (*inter alia*) **Fig. 3** and the corresponding text on p. 20, lines 3 - 7. With regard to the “means for separating” element of independent Claims 1, 13, and 39, refer to (*inter alia*) **Figs. 3 - 4** and the corresponding text on p. 20, lines 7 - 11. The “means for using” element of independent Claims 1, 13,and 39 is described by (*inter alia*) the text on p. 20, lines 11 - 15.

11. The “means for using” element in dependent Claims 12 and 24 is described, for example, by text on p. 20, lines 15 - 17.

## 6) GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

12. The first ground of rejection presented for review is a rejection of Claim 52 as failing to

comply with the written description requirement of 35 U.S.C. §112, first paragraph.

13. The second ground of rejection presented for review is a rejection of Claims 13 - 14, 18 - 19, 21 - 22, 24 - 26, 30, 32, 34 - 35, 37, 39 - 40, 44, 47, and 49 - 52 as being anticipated under 35 U.S.C. §102(b) by Patel et al ("An Efficient Discrete Log Pseudo Random Generator", hereinafter referred to as "Patel").

14. The third ground of rejection presented for review is a rejection of Claims 1 - 2, 6 - 7, 9 - 12, 23, 36, and 48 as being unpatentable over Patel in view of Schneier, "Applied Cryptography".

## 7) ARGUMENT

### 7.1) First Ground of Rejection

15. Page 5, lines 22 - 25 of the Office Action dated February 15, 2006 (hereinafter, "the Office Action") states that Claim 52 is rejected under 35 U.S.C. §112, first paragraph, as failing to comply with the written description requirement. Claim 52 is an independent claim.

16. The Office Action further states (on p. 6, lines 1 - 4) that "the specification does not provide antecedent basis for setting (N-C) uppermost **contiguous ones of bits** to zero and leaving the lowermost **contiguous ones of bits** random" (emphasis original). Appellant respectfully disagrees.

17. The Office Action does not specify which lines of Claim 52 are referred to, and Appellant

notes that this claim does not specify a "setting ..." element. Appellant presumes the Examiner is referring to either Claim 52, lines 3 - 4 (which pertain to the seed value that is used on a first iteration of the generator) or Claim 52, lines 11 - 13 (which pertain to subsequent iterations of the generator). Furthermore, it is unclear whether the Examiner objects to the term "random", the term "contiguous", or the terms "uppermost" and "lowermost". Accordingly, each of these cases will now be discussed.

18. Appellant respectfully notes that only the claim language on lines 3 - 4 of Claim 52 uses the terminology "... bits are random" (emphasis added), when referring to the seed used with the generator (i.e., the input for the first iteration of the generator); other limitations (which pertain to subsequent iterations of the generator) refer to pseudo-random bits. Use of random bits for the first iteration is described on p. 2, lines 6 - 7 ("A pseudo-random number generator ... typically takes as input a relatively short random string ..."); and p. 6, lines 10 - 12 ("... hardware generators are typically used to produce the short random seeds which are then provided as input to a PRBG [pseudo-random bit generator]...". Furthermore, see Specification, p. 16, lines 10 - 11, stating that the seed is of length "C" bits, and p. 16, line 12, stating that "the top (N - C) bits of each iteration [which includes the first iteration] are set to all zeroes" (emphasis added).

19. Accordingly, Appellant respectfully submits that his specification provides clear support for use of random bits for the seed, where the number of these random bits is some number "C", while the (remaining) top (N - C) bits are "all set to zeroes".

20. Setting of <u>contiguous</u> bits (where this "contiguous" claim language is used in Claim 52, lines 3 - 4 and lines 11 - 13) is discussed in several places within Appellant's specification. See, for example, Specification p. 9, lines 8 - 9, stating "The subset of bits [which are selected as a next sequential input value for the generator; p. 9, lines 3 - 4] <u>may be a contiguous group of bits</u>, or a non-contiguous group of bits" (emphasis added). See also original Claims 10, 22, and 35, specifying "... selecting the subset of bits comprises selecting a <u>contiguous</u> group of bits" (emphasis added). In addition, the text on p. 20, lines 7 - 10 states that the 160 bits (where 160, in this example, is the value of "C") which are selected as the input to the next iteration of the generator "are not required to be the top-most bits, nor are they required to be contiguous"; Claim 52, by contrast, specifies a limitation where the bits <u>are required</u> to be <u>contiguous</u>. See also Specification, p. 16, line 12, "the top (N - C) bits of each iteration <u>are set to all zeroes</u>" (emphasis added). It is clear that the "the top (N - C) bits" refers to (N - C) <u>contiguous</u> bits.

21. Using terms well known to those in the art, the "top" (N - C) bits of a number may be alternatively referred to as the "uppermost" (N - C) bits of the number. Accordingly, if "the <u>top</u> (N - C) bits" <u>of an N-bit number</u> are set to zeroes, then there are "C" bits <u>remaining</u> in this <u>N-bit number</u>; and, since the (N - C) bits are the <u>top/uppermost</u> bits, then the "C" bits are the <u>lowermost</u> bits of the N-bit number.

22. Accordingly, Appellant respectfully submits that his specification provides clear support for use of <u>contiguous</u> bits, both for the (N - C) bits that are set to zeroes and for the "C" bits that are used as the actual exponent for the generator. Furthermore, Appellant respectfully submits

that his specification provides clear support for the terms uppermost and lowermost when describing the all-zero bits and the other "C" bits, respectively (whether those "C" bits are the random bits of the seed used in the first iteration or the generated pseudo-random bits of other iterations, as discussed above in paragraph 18).

23. In view of paragraphs 17 - 22, Appellant respectfully submits that the §112, first paragraph rejection of Claim 52 is improper. (And regarding the §112, second paragraph rejection discussed on p. 6, lines 11 - 20 of the Office Action, Appellant apologizes for any confusion caused by this term, and has no objection to deletion of "effectively-short" from Claim 52, line 6.)

**7.2) Second Ground of Rejection**

24. Appellant respectfully submits that a *prima facie* case of anticipation under 35 U.S.C. §102 has not been made out as to Claims 13 - 14, 18 - 19, 21 - 22, 24 - 26, 30, 32, 34 - 35, 37, 39 - 40, 44, 47, and 49 - 52. Section 706.02 of the MPEP, "Rejection on Prior Art", states in Section IV, "Distinction Between 35 U.S.C. 102 and 103", the requirements for establishing a *prima facie* case of anticipation under this statute, noting that "... for anticipation under 35 U.S.C. 102, the reference must teach every aspect of the claimed invention either explicitly or impliedly" (emphasis added). This requirement is also stated in MPEP §2131, "Anticipation -- Application of 35 U.S.C. 102(a), (b), and (e)", which states (in its final paragraph) "A claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described, in a single prior art reference", quoting *Verdegaal Bros. v. Union Oil Co. of*

*California*, 814 F.2d 628, 631, 2 USPQ2d 1051, 1053 (Fed. Cir. 1987), emphasis added. This final paragraph of MPEP §2131 also states "The elements <u>must be arranged</u> as required by the claim ...", quoting *In re Bond*, 910 F.2d 831, 15 USPQ2d 1566 (Fed. Cir. 1990), emphasis added.

25. Furthermore, Appellant is entitled to have <u>all words</u> of his claimed invention considered when determining patentability. See Section 2143.03 of the MPEP, "All Claim Limitations Must Be Taught or Suggested", referencing ***In re Wilson***, 165 USPQ 494, 496 (C.C.P.A. 1970), which stated "*All words* in a claim must be considered in judging the patentability of that claim against the prior art." (emphasis added).

26. The burden for rebutting a rejection under 35 U.S.C. 102 does not pass to Appellant until a *prima facie* case of anticipation has been made out. See *In re Bass*, 177 USPQ 178, 186 (C.C.P.A. 1973), which held:

> From the evidence available to it, the initial burden of making out a prima facie case of prior invention is on the Patent Office. . . . When the Patent Office has made out a prima facie case of priority the burden would then shift to the applicant to rebut it.

Accordingly, Appellant respectfully submits that the burden has not passed. For the sake of expediency, Appellant will, however, provide a rebuttal herein of the analysis provided in the Office Action.

### 7.2.1) Rejection of Independent Claim 52

27. Referring first to independent Claim 52, Appellant respectfully submits that Patel fails to teach all limitations of this claim, and in particular, does not teach "each and every element" or "all words" of this claim. The Office Action analysis therefore fails to make out a *prima facie* case of anticipation, in violation of the above-quoted MPEP §706.02, §2131, and §2143.03, as will now be demonstrated.

28. Page 10, line 14 - p. 11, line 9 of the Office Action analyze Appellant's independent Claim 52. This analysis will now be described.

29. Regarding the first element of Claim 52, "providing an N-bit input value in which (N - C) uppermost contiguous ones of the bits are all set to zeroes and in which C lowermost contiguous ones of the bits are random" (Claim 52, lines 3 - 4, emphasis added), the Office Action cites p. 313, lines 22 - 27 of Patel (Office Action, p. 10, line 17). Appellant respectfully disagrees with this analysis, and notes that the cited text specifies that the lowermost bits "are known to be zero" (... all the bits to the right of the shifted *s* are known to be zero"; Patel, p. 313, line 23 - 24, emphasis added). Having the lowermost bits set to zero, as specified by Patel, is patentably distinct from an input value "in which C lowermost contiguous ones of the bits are random" (Claim 52, line 4, emphasis added). Accordingly, Appellant respectfully submits that Patel does not teach this claim element. Furthermore, Appellant notes that the Office Action admits that Patel uses the "trailing" bits for output (Office Action, p. 11, line 3) and uses the "leading" bits for the $\omega(\log n)$ bits (Office Action, p. 11, line 3); this is the opposite approach to Appellant's claim language, where the "C" bits are the lowermost bits (Claim 52, lines 4) and the (N-C) uppermost

bits are set to zeroes (Claim 52, line 3 - 4).

30. Page 313, lines 22 - 27 of Patel are also cited (Office Action, p. 11, line 6) as teaching the element on lines 11 - 13 of Appellant's Claim 52 ("creating a new N-bit input value in which the (N - C) <u>uppermost</u> contiguous ones of the bits are all set to zeroes and in which the <u>lowermost</u> C contiguous ones of the bits are set to the <u>C-bit portion</u>"; emphasis added). As demonstrated above in paragraph 29, Patel specifies that the <u>lowermost</u> bits are set to <u>zero</u>. Furthermore, Appellant notes that Patel specifies that *s* is "shift[ed] ... to the <u>left</u> by squaring *S* the appropriate number of times" (Patel, p. 313, lines 2 - 3, emphasis added), after which "all the bits to the <u>right</u> of the $i^{th}$ bit are zero" (Patel, p. 313, lines 3 - 4, emphasis added). Accordingly, Patel's lowermost bits are zero; this is distinct from Appellant's claim language, where the <u>uppermost</u> (N - C) contiguous ones of the bits -- rather than Patel's <u>lowermost</u> bits -- are set to zeroes (Claim 52, lines 11 - 12, "the (N - C) <u>uppermost</u> contiguous ones of the bits are all set to zeroes"; emphasis added). Furthermore, as discussed in detail below in paragraphs 39 - 43, this text on Page 313 of Patel <u>does not pertain</u> to how his <u>generator</u> function works; instead, it is a discussion of an <u>inverse</u> function, and is provided with regard to Theorem 9 on p. 312 (where Theorem 9 attempts to disprove the security of the Patel generator).

31. As demonstrated by paragraphs 29 - 30 herein, Appellant respectfully submits that the Office Action fails to cite a reference that teaches <u>each and every element</u> of Appellant's independent Claim 52, and fails to cite a reference that teaches <u>all words</u> of the claim language of this claim.

32. Appellant therefore respectfully submits that the Office Action fails to make out a *prima facie* case of anticipation as to independent Claim 52, in violation of the above-quoted MPEP §706.02, §2131, and §2143.03. Without more, this claim is deemed patentable. See ***In re Oetiker***, 24 USPQ 2d 1443, 1444 (Fed. Cir. 1992), which stated:

> If the examination at the initial stage does not produce a prima facie case of unpatentability, then without more the applicant is entitled to grant of the patent.

**7.2.2) Rejection of Independent Claims 13, 25, and 39**

33. Referring now to independent Claims 13, 25, and 39, Appellant respectfully submits that Patel <u>fails to teach</u> all limitations of these claims, and in particular, does <u>not</u> teach "each and every element" or "all words" of these claims. The Office Action analysis therefore fails to make out a *prima facie* case of anticipation, in violation of the above-quoted MPEP §706.02, §2131, and §2143.03, as will now be demonstrated.

34. With regard to Appellant's independent Claims 13, 25, and 39, the Office Action cites p. 313, section 5 of Patel as well as section 7.1 on p. 316 (Office Action, pages 7 - 9). Appellant's independent Claims 13, 25, and 39 explicitly specify "an input value comprising <u>C random bits</u>" (Claim 13, line 3, emphasis added), which is provided "as a <u>short exponent</u> x of a 1-way function G**x mod P" (Claim 13, lines 4 - 5, emphasis added). This 1-way function "generat[es] an output sequence comprising N pseudo-random bits" (Claim 13, lines 4 - 5). The N bits are "separat[ed] ... into a C-bit portion and an (N-C)-bit portion" (Claim 13, lines 8 - 9). The C-bit portion is used "as the provided input value for [i.e., as an *exponent* of; see Claim 13, line 5] a

next iteration of [the function]" (Claim 13, lines 10 - 11) and the (N-C)-bit portion is used "as pseudo-random output bits" of the generator (Claim 13, lines 11 - 13).

35. Appellant respectfully submits that the "NEW GENERATOR" described in Patel's Section 5 (see p. 313, last paragraph) describes use of an N-bit exponent, as has been discussed in detail in Appellant's prior response transmitted on June 6, 2005 (hereinafter, "Appellant's June, 2005 response), which is hereby incorporated herein by reference. See, also, p. 316, Section 7.1, lines 1 - 6 of Patel, stating "Let us focus on the mechanics of the generator [described in the Patel paper] ... each iteration [of this generator] involves a large exponent" (emphasis added). Patel's use of an N-bit exponent, referred to in Section 7.1 as a large exponent, is patentably distinct from Appellant's claimed use of a short, C-bit exponent.

36. Furthermore, Appellant respectfully submits that while Patel discusses producing "n - c" bits per iteration of his generator, there is no teaching in Patel that any "c" of those bits are used as the exponent in a next-sequential iteration of the generator, in contrast to Appellant's claimed invention, which specifies "... using the C-bit portion ... as the provided input value for a next iteration ..." (Claim 13, lines 10 - 11), where this "provided input value" is used "as a short exponent ..." (Claim 13, lines 4 - 5).

37. In fact, rather than using "C" of the generated pseudo-random bits as the next, short exponent, as claimed by Appellant, Patel states in Section 7.1, lines 7 - 13 that -- if the exponent was to be restricted to a short exponent (see Section 7.1, line 12, "when we restrict our

exponents") -- "we no longer have a permutation. Hence the simple construction used here [i.e., the simple construction of a generator G^($x_i$) mod P] is inapplicable" (emphasis added). Patel then states that a possible method of "settling this problem" that arises through use of a short exponent is to construct a "perfect extender" and then achieve the pseudo-random generation "through repeated applications of the extender to a random seed" (Patel, Section 7, lines 13 - 18, emphasis added). Appellant's claimed invention does not repeatedly use the random seed. Rather, the C-bit random seed is specified only in the first element of independent Claims 13, 25, and 39 ("an input value comprising C random bits"), and it is clear that subsequent iterations of Appellant's generator do not reuse this random seed, but instead use C bits that are newly generated from each iteration of the generator. See Claim 13, lines 8 - 9, "... separating the N bits of the [newly] generated N-bit output sequence into a C-bit portion ...", and lines 10 - 11, "... using the [newly-generated] C-bit portion of the generated N-bit output sequence as the provided input value for a next iteration ..." (emphasis added). In other words, in Appellant's claimed invention, the bits used for the exponent (except for the first iteration) are "of" (i.e., taken directly from) the generated N-bit output sequence; because the bits are taken from the generated output sequence, this necessarily implies that the bits in Appellant's claimed approach are not modified using a perfect extender or by reusing a random seed (or by any other type of modifying function) in between iterations of the generator, in contrast to Patel's "possible method of settling this problem" by repeatedly applying a perfect extender to a random seed.

38. Accordingly, whereas Patel teaches that a perfect extender is used, achieving pseudo-random generation by repeatedly applying the extender to a random seed (Patel, Section 7.1, lines

13 - 18), Appellant's claimed invention does not use this approach and is therefore patentably distinct.

39. In Section 5.1 of Patel, there is a discussion of use of a short exponent (in contradiction to the statement in Section 7.1, lines 5 - 6, "... each iteration involves a large exponent"; emphasis added). Appellant respectfully submits that this Section 5.1 does not pertain to Patel's generator but rather pertains to his proof of security of that generator. See, for example, the Title of Section 5.1, "Proof of Security", on p. 314 of Patel. See also the second paragraph of this section, stating "Now we note that we can use this [i.e., a weak oracle] to discover $s$ given $g^s$ mod $p$ where s has $\omega(\log n)$ bits. ... Thus we can discover the $i^{[th]}$ bit [generated by the generator] in poly(n) time". Accordingly, Patel is discussing how to "break" the security of the generator, thereby discovering "s" when given the value f(s). See, for example, Appellant's specification p. 4, lines 3 - 6, which refer to this technique of "determin[ing] the value of x given the value of f(x)"; note also that Patel's reference to "discover[ing] $s$ given $g^s$ mod p" is another way of saying "discover $s$ given f(s), where f(s) = $g^s$ mod p". Accordingly, in Section 5.1, Patel is describing how to use output values created by his generator (using, in "each iteration ... a large exponent"; Section 7.1, lines 5 - 6) -- which are supposedly pseudo-random values -- to discover the input values that were used to generate those created output values. This is a discussion of the discrete logarithm function, and not of Patel's generator.

40. See also Patel's Theorem 9, referenced at Section 5.1, lines 9 - 10. Note that Theorem 9 (page 312) discusses $n - \omega(\log n)$ bits of the output of Patel's generator -- that is, "$g^x$ mod p" is

an output of the generator, and thus "trailing bits of $g^x$ mod p" are bits of the generated output. The discussion of Theorem 9 on p. 312 - 313 discusses how to predict the ith bit of an input value x, given the value $g^x$ that has been generated using that input value. Again, this is not a discussion of Patel's generator, but rather it is a discussion of whether it can be proven that the generator is secure; if it is secure, then it will be computationally infeasible to predict the ith bit of the input value x. (See, for example, p. 2, lines 12 - 13 and p. 4, line 4 - p. 5, line 2 of Appellant's specification, which discuss "secure" generators as being those that are computationally infeasible to solve.) Patel continues, in the discussion of Theorem 9, by focusing (by way of example) on a generator that uses a short exponent (page 313, lines 1 - 2, "Now pick an element $S = g^s$ where s is a short exponent"). Patel then discusses whether any of the *i* bits can be predicted in this simpler, "short exponent", scenario, whereby the short exponent can be manipulated by shifting its bits. This discussion continues by discussing various manipulations of the short exponent, whereby the bits are manipulated such that a bit in the ith position can be discovered repeatedly, until all of the bits of the input exponent *s* have been "recovered" (p. 313, lines 16 - 17). After recovering/guessing all of these bits, there is a "candidate" for the input exponent *s*, and this candidate is used as an input exponent for the generator; if the output of this computation (g**candidate) is equal to the value S that was previously computed with the generator, then the security of the generator is disproven because the input value has been deduced from the output value.

41. Accordingly, this discussion of "short exponents" with regard to Patel's Theorem 9 for proving security of Patel's generator (which proof of security is then subsequently addressed in

Section 5.1) is not a statement that Patel's actual generator uses short exponents. Rather, short exponents are used in the discussion of Theorem 9 on p. 312 - 313 as a simplification during the proof of security exercise being conducted therein for Patel's generator.

42. Stated another way, in Patel's paper, the problem addressed in Theorem 9 and Section 5.1 is to solve Y = G**S where S is a C-bit number. That is, given an output value Y and a generator G that was used to compute that value Y, the problem is to find the input value S from which the output value Y was computed. If the input value S can be found, given the output value Y, then the function is not a 1-way function and is not secure. (As will be obvious, if S is a full N-bit number, in contrast to Patel's teaching of using a shorter C-bit number in the proof of security, then solving for an N-bit S is even more difficult than solving for a shorter C-bit value of S. Thus, Patel explicitly makes the assumption about using shorter exponents, for convenience, during his proof of security.)

43. Accordingly, if the input value can be recovered when given the output value -- according to Patel's Theorem 9 -- then the output value is not pseudo-random. This is in contrast to Appellant's claimed invention, which specifies that N pseudo-random bits are generated when using a short, C-bit exponent (Claim 13, line 4). And as noted above in paragraph 42, the ability to recover the input value, using Patel's approach, further establishes that the generator described therein is not a 1-way function and is not secure. This is in contrast to Appellant's claimed invention, which specifies a secure, 1-way function (Claim 13, line 5).

44. Appellant's independent Claims 13, 25, and 39 further specify that an (N-C)-bit portion of the generated N-bit pseudo-random output sequence is used "as pseudo-random output bits" (Claim 13, lines 11 - 12). Patel similarly discusses outputting "$n$ - ω(log $n$) bits". See p. 313, final sentence discussing "NEW GENERATOR", and Section 7.1, final sentence of first paragraph ("The output of the generator are the trailing $n$ - ω(log $n$) bits of $x_i$ ..."). Appellant's Claims 13, 25, and 39 specify "using the C-bit portion of the generated N-bit output sequence [of a just-completed iteration] as the provided input value [i.e., as a <u>short</u> exponent; see Claim 13, line 5] for a next iteration of the [generator]" (Claim 13, lines 10 - 11). By contrast, Patel's NEW GENERATOR uses $n$ - ω(log $n$) bits as output but teaches using <u>all N</u> of the bits as the exponent. Refer again to Appellant's June, 2005 response, where this N-bit exponent was discussed in detail.

45. Appellant also respectfully submits that Patel <u>admits</u> that he does not know how to make his generator work if <u>only C bits</u> are used as the <u>exponent</u>. See paragraph 37, above, discussing the statement in Patel "Hence the simple construction used here [i.e., the simple construction of a generator G^($x_i$) mod P] <u>is inapplicable</u>" (emphasis added)", referring to Section 7.1, lines 12 - 13. In view of this admission by Patel, Appellant respectfully submits that Patel's text is <u>not enabling</u>. If a reference is <u>not enabling</u>, then it does <u>not</u> qualify as prior art for a §102 rejection. See ***In re Sun***, 31 USPQ 2d 1451, 1453 (Fed. Cir. 1993) (unpublished), which stated

> But to be prior art under section 102(b), a reference <u>must be enabling</u>. . . . That is, it must put the claimed invention in the hand of one skilled in the art. (emphasis added)

See also ***Akzo N.V. v. United States International Trade Commission***, 1 USPQ 2d 1241, 1245 (Fed. Cir. 1986), ***cert. denied***, 482 U.S. 909 (1987), which stated

> Under 35 U.S.C. §102, anticipation requires that each and every element of the claimed invention be disclosed in the prior art. . . . In addition, the prior art reference must be enabling, thus placing the allegedly disclosed matter in the possession of the public. (emphasis added)

Because Patel's discussion of using a short exponent with a generator is not enabling, and therefore cannot be used as prior art under §102, Appellant respectfully submits that Patel cannot be used to anticipate his claimed invention.

46. Because the Office Action fails to cite a reference that teaches each and every element of Appellant's independent Claims 13, 25, and 39, and fails to cite a reference that teaches all words of this claim language, arranged as required by Appellant's claim language, as demonstrated by paragraphs 34 - 45, Appellant respectfully submits that the §102 rejection fails to meet the requirements of Sections 706.02 and 2131 of the MPEP (which were noted above in paragraph 24). The Office Action therefore fails to make out a *prima facie* case of anticipation as to these independent claims. Without more, these claims are deemed patentable.

### 7.2.3) Rejection of Dependent Claims 14, 18 - 19, 21 - 24, 26, 30, 32, 34 - 37, 40, 44, 48 - 51

47. Dependent Claims 14, 18 - 19, 21 - 24, 26, 30, 32, 34 - 37, 40, 44, and 48 - 51 stand or fall with independent Claims 13, 25, and 39, from which they depend. Thus, these claims are deemed allowable by virtue of the allowability of independent Claims 13, 25, and 39, as

established in Section 7.2.2 herein.

## 7.3) Third Ground of Rejection

48. Page 11, lines 20 - 22 of the Office Action state that Claims 1 - 2, 6 - 7, 9 - 12, 23, 36, and 48 are rejected under 35 U.S.C. §103(a) as being unpatentable over Patel in view of Schneier, "Applied Cryptography". Claim 1 is an independent claim. The remaining ones of these claims are dependent claims.

49. Appellant respectfully submits that a *prima facie* case of obviousness under 35 U.S.C. §103 has not been made out as to his Claims 1 - 2, 6 - 7, 9 - 12, 23, 36, and 48. Section 706.02(j) of the MPEP, "Contents of a 35 U.S.C. 103 Rejection", states the requirements for establishing a *prima facie* case of obviousness under this statute, noting that three criteria must be met. These criteria are (1) a suggestion or motivation, found either in the references or in the knowledge generally available, to modify or combine the references; (2) a reasonable expectation of success; and (3) the combination must teach or suggest all the claim limitations. This text goes on to state that "The initial burden is on the examiner to provide some suggestion of the desirability of doing what the inventor has done.". The three requirements for establishing a *prima facie* case of obviousness are also stated in MPEP §2142, "Legal Concept of *Prima Facie* Obviousness", and MPEP §2143, "Basic Requirements of a *Prima Facie* Case of Obviousness".

### 7.3.1) Rejection of Independent Claim 1

50. Regarding the §103 rejection of independent Claim 1, Appellant respectfully submits that

elements of this claim are distinct from Patel for the same reasons discussed above in **Section 7.2.1** with regard to independent Claims 13, 25, and 39 (given that independent Claims 13, 25, and 39 specify limitations analogous to those of independent Claim 1).

51. Accordingly, Appellant respectfully submits that the Office Action fails to make out a *prima facie* case of unpatentability as to independent Claim 1, and without more, this claim is deemed patentable.

### 7.3.2) Rejection of Dependent Claims 2, 6 - 7, 9 - 12, 23, 36, and 48

52. Dependent Claims 2, 6 - 7, 9 - 12, and 48 stand or fall with independent Claim 1, from which they depend. Thus, these claims are deemed allowable by virtue of the allowability of Claim 1, the patentability of which is discussed above in paragraphs 50 - 51. Dependent Claim 23 stands or falls with independent Claim 13, from which it depends, and dependent Claim 36 stands or falls with independent Claim 25, from which it depends. The patentability of these independent claims is discussed above in paragraphs 33 - 46.

## 8) CONCLUSION

For the reasons set out above, Appellant respectfully contends that each appealed claim is patentable, and respectfully request that the Examiner's Final Rejection of appealed Claims 1 - 2, 6 - 7, 9 - 14, 18 - 19, 21 - 26, 30, 32, 34 - 37, 39 - 40, 44, and 47 - 52 should be reversed.

Respectfully submitted,

/Marcia L. Doubet/ /#40,999/

Marcia L. Doubet,
Attorney for Appellant
Reg. No. 40,999

Customer Number for Correspondence: 43168
Phone: 407-343-7586
Fax: 407-343-7587

## CLAIMS APPENDIX

CLAIMS AS CURRENTLY PRESENTED:

Claim 1: A computer program product for efficiently generating pseudo-random bits, the computer program product embodied on one or more computer readable media and comprising:

computer-readable program code means for providing an input value comprising C random bits;

computer-readable program code means for generating an output sequence comprising N pseudo-random bits using the provided C-bit input value as a short exponent x of a 1-way function G**x mod P that comprises modular exponentiation modulo a safe N-bit prime number P, wherein a base G of the modular exponentiation is a fixed generator value;

computer-readable program code means for separating the N bits of the generated N-bit output sequence into a C-bit portion and an (N-C)-bit portion; and

computer-readable program code means for using the C-bit portion of the generated N-bit output sequence as the provided input value for a next iteration of the computer-readable program code means for generating while using the (N-C)-bit portion of the generated N-bit output sequence as pseudo-random output bits, until a desired number of pseudo-random output bits have been generated.

Claim 2: The computer program product according to Claim 1, wherein the 1-way function is based upon an assumption known as "the discrete logarithm with short exponent" assumption.

Claims 3 - 5 (canceled)

Claim 6: The computer program product according to Claim 1, wherein C = 160 and N = 1024.

Claim 7: The computer program product according to Claim 1, wherein C is greater than or equal to 160 and N is greater than or equal to 1024.

Claim 8 (canceled)

Claim 9: The computer program product according to Claim 1, wherein the (N-C)-bit portion is concatenated to pseudo-random output bits previously generated by the computer-readable program code means for generating.

Claim 10: The computer program product according to Claim 1, wherein the (N-C)-bit portion is a contiguous group of (N-C) bits from the generated N-bit output sequence.

Claim 11: The computer program product according to Claim 1, wherein the (N-C)-bit portion is a non-contiguous group of (N-C) bits from the generated N-bit output sequence.

Claim 12: The computer program product according to Claim 1, further comprising computer-readable program code means for using the desired number of generated pseudo-random bits as input to an encryption operation.

Claim 13: A system for efficiently generating pseudo-random bits in a computing environment, comprising:

means for providing an input value comprising C random bits;

means for generating an output sequence comprising N pseudo-random bits using the provided C-bit input value as a short exponent x of a 1-way function G**x mod P that comprises modular exponentiation modulo a safe N-bit prime number P, wherein a base G of the modular exponentiation is a fixed generator value;

means for separating the N bits of the generated N-bit output sequence into a C-bit portion and an (N-C)-bit portion; and

means for using the C-bit portion of the generated N-bit output sequence as the provided input value for a next iteration of the means for generating while using the (N-C)-bit portion of the generated N-bit output sequence as pseudo-random output bits, until a desired number of pseudo-random output bits have been generated.

Claim 14: The system according to Claim 13, wherein the 1-way function is based upon an assumption known as "the discrete logarithm with short exponent" assumption.

Claims 15 - 17 (canceled)

Claim 18: The system according to Claim 13, wherein C = 160 and N = 1024.

Claim 19: The system according to Claim 13, wherein C is at least 160 and N is at least 1024.

Claim 20 (canceled)

Claim 21: The system according to Claim 13, wherein the (N-C)-bit portion is concatenated to pseudo-random output bits previously generated by the means for generating.

Claim 22: The system according to Claim 13, wherein the (N-C)-bit portion is a contiguous group of (N-C) bits from the generated N-bit output sequence.

Claim 23: The system according to Claim 13, wherein the (N-C)-bit portion is a non-contiguous group of (N-C) bits from the generated N-bit output sequence.

Claim 24: The system according to Claim 13, further comprising means for using the desired number of generated pseudo-random output bits as input to an encryption operation.

Claim 25: A programmatic method for efficiently generating pseudo-random bits, comprising the steps of:

providing an input value comprising C random bits;

generating an output sequence comprising N pseudo-random bits using the provided C-bit input value as a short exponent x of a 1-way function G**x mod P that comprises modular exponentiation modulo a safe N-bit prime number P, wherein a base G of the modular exponentiation is a fixed generator value;

separating the N bits of the generated N-bit output sequence into a C-bit portion and an (N-C)-bit portion; and

using the C-bit portion of the generated N-bit output sequence as the provided input value for a next iteration of the generating step while using the (N-C)-bit portion of the generated N-bit output sequence as pseudo-random output bits, until a desired number of pseudo-random output bits have been generated.

Claim 26: The method according to Claim 25, wherein the 1-way function is based upon an assumption known as “the discrete logarithm with short exponent” assumption.

Claims 27 - 29 (canceled)

Claim 30: The method according to Claim 25, wherein C = 160 and N = 1024.

Claim 31 (canceled)

Claim 32: The method according to Claim 25, wherein C is greater than or equal to 160 and N is greater than or equal to 1024.

Claim 33 (canceled)

Claim 34: The method according to Claim 25, wherein the (N-C)-bit portion is concatenated to pseudo-random output bits previously generated by the generating step.

Claim 35: The method according to Claim 25, wherein the (N-C)-bit portion is a contiguous group of (N-C) bits from the generated N-bit output sequence.

Claim 36: The method according to Claim 25, wherein the (N-C)-bit portion is a non-contiguous group of (N-C) bits from the generated N-bit output sequence.

Claim 37: The method according to Claim 25, further comprising the step of using the desired number of generated pseudo-random output bits as input to an encryption operation.

Claim 38 (canceled)

Claim 39: An encryption system, comprising:

means for providing an input value comprising C random bits;

means for generating an output sequence comprising N pseudo-random bits using the provided C-bit input value as a short exponent x of a 1-way function G**x mod P that comprises modular exponentiation modulo a safe N-bit prime number P, wherein a base G of the modular exponentiation is a fixed generator value;

means for separating the N bits of the generated N-bit output sequence into a C-bit portion and an (N-C)-bit portion;

means for using the C-bit portion of the generated N-bit output sequence as the provided input value for a next iteration of the means for generating while using the (N-C)-bit portion of the generated N-bit output sequence as pseudo-random output bits, until a desired number of pseudo-random output bits have been generated; and

means for using the desired number of generated pseudo-random bits as input to an encryption operation.

Claim 40: The encryption system according to Claim 39, wherein the 1-way function is based upon an assumption known as "the discrete logarithm with short exponent" assumption.

Claims 41 - 43 (canceled)

Claim 44: The encryption system according to Claim 39, wherein C = 160 and N = 1024.

Claims 45 - 46 (canceled)

Claim 47: The encryption system according to Claim 39, wherein the (N-C)-bit portion is concatenated to pseudo-random output bits previously generated by the means for generating.

Claim 48: The computer program product according to Claim 1, wherein N is greater than or equal to (C * 6).

Claim 49: The system according to Claim 13, wherein N is greater than or equal to (C * 6).

Claim 50: The method according to Claim 25, wherein N is greater than or equal to (C * 6).

Claim 51: The encryption system according to Claim 39, wherein N is greater than or equal to (C * 6).

Claim 52: A programmatic method for efficiently generating pseudo-random bits, comprising the steps of:

providing an N-bit input value in which (N-C) uppermost contiguous ones of the bits are all set to zeroes and in which C lowermost contiguous ones of the bits are random;

generating an output sequence comprising N pseudo-random bits using the provided N-bit input value as an effectively-short, C-bit exponent x of a 1-way function G**x mod P that comprises modular exponentiation modulo a safe N-bit prime number P, wherein a base G of the modular exponentiation is a fixed generator value;

separating the N bits of the generated N-bit output sequence into a C-bit portion and an (N-C)-bit portion;

creating a new N-bit input value in which the (N-C) uppermost contiguous ones of the bits are all set to zeroes and in which the lowermost C contiguous ones of the bits are set to the C-bit portion; and

using the new N-bit input value as the provided input value for a next iteration of the generating step while using the (N-C)-bit portion of the generated N-bit output sequence as pseudo-random output bits, until a desired number of pseudo-random output bits have been generated.

## EVIDENCE APPENDIX

Appellant, the Appellant' legal representative, and the assignee have no personal knowledge of evidence requiring separate identification herein as bearing on this Appeal.

## RELATED PROCEEDINGS APPENDIX

No related proceedings are personally known to Appellant, the Appellant's legal representative, or the assignee.